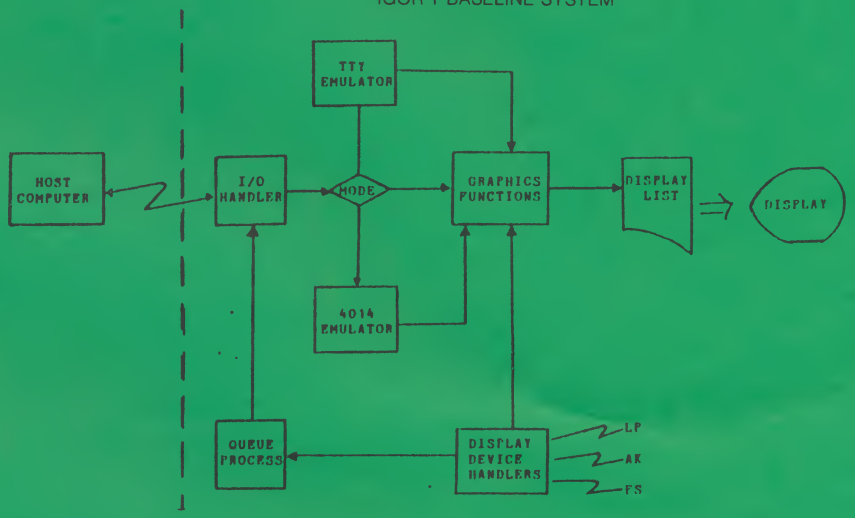
Interactive Graphic Systems Inc.

IGOR 1

IGOR 1 is a multi-faceted program that allows the graphics hardware to be utilized as an intelligent work station, while allowing it to operate as a graphic satellite to a host computer. IGOR is ideally suited for distributed graphic processing systems. One of IGOR's best attributes is that it off-loads much of the graphic processing normally handled by hosts. The savings to the host can be up to 90% of the overhead incurred on a normal graphic system. The TTY mode allows the station to communicate to any computer system that uses an RS232C serial interface, or IGOR 1 may be configured to use a DMA interface, whichever suits the application needs. IGOR 1 is designed to be a base for future advanced IGOR systems. IGOR 1 handles all graphics functions to which present IGS graphic users have become accustomed. IGOR 1 meets the needs of the satellite environment without sacrificing the speed and efficiency that IGS software has always had. Not only is IGOR 1 designed for the end-user market but both the CAD/CAM and military markets will find IGOR 1 attractive. To meet these special needs IGOR 1 is designed in modular format and programmed utilizing structural programming techniques. Because of these approaches IGOR 1 can accommodate the inclusion of new devices, handlers for different types of interfaces, and can be easily modified. However, the majority of users will find IGOR 1 satisfactory in its current configuration.

IGOR 1 BASELINE SYSTEM



GRAPHICS CONTROL
- Dynamic buffer management
- Read from/write to buffer
- Attention control
- Multiple display/station control
- Text read/write

TTY MODE
- Use GRAPHICS STATIONS as TTY terminals
- 56 lines × 102 columns

4014 MODE (OPTIONAL)
- Emulates Tektronix 4014 graphics terminal
- Allows PLOT10 usage

HOST INTERFACE
- RS232C 0-19.2K baud
- DMA-DR11B

IGOR 1 OPERATING SYSTEM
- Process control functions
- Time slicing
- Multiple priority levels
- Dynamic memory allocation/de-allocation
- Event driven
- Easily modified to meet OEM and application needs

IGS DEBUGGER
- Relative addressing
- Multiple breakpoints
- Access to all registers
- Single/auto step
- Memory dump/modify

P.O. Box 1305/Woodland Hills, California 91365/(213) 346-0549/Twx 910-494-5986

IGS

Interactive Graphic Systems Inc.

GAM

GRAPHIC ACCESS METHOD (GAM)
GAM is the graphics input/output handler that provides graphics programming services for the assembler language progammer. GAM allows full interaction between a program and a graphic station. A station is referenced from the graphics program as any other peripheral device. GAM's purpose is to control the man-machine interaction in the display environment, and to provide system integrity without impeding station interaction. GAM achieves this purpose by functioning as a handler, rather than as a modification to the operating system. GAM is designed to run on a PDP-11 or VAX computer under any of the following operating systems:

- ☐ IAS
- ☐ RSX 11M/M+
- ☐ VMS

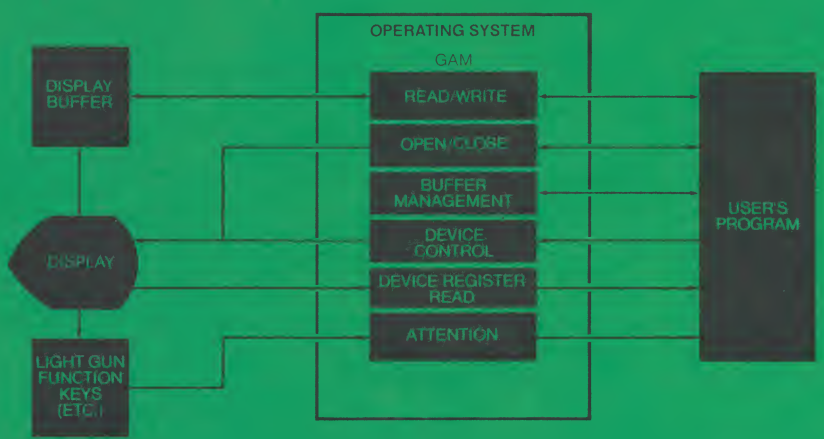
The programmer interface to GAM for each of these systems is identical. The compatability between systems is made possible by a set of MACROs provided with GAM. In addition to the GAM handler MACROs, a set of some 90+ MACROs is provided which aids the assembly programmer in building graphic display lists. These GAM MACROs are usually added to the System MACRO Library (SML) The GAM services include:

- Input/output MACROs to provide the means to read and write to the display buffer. Other MACROs to read station registers.
- Station control MACROs that provide the programmer with the capability to start and stop the station.
- Buffer management facilities for dynamically assigning and releasing buffer space according to program need while providing mutual protection against intrusion and sharing display buffer space.
- Facilities for attention handling (the display equivalent of computer interrupts) are under program control. The true value of graphics lies in its unique capability to produce sophisticated lines of communication between the station and the application program. GAM provides a set of MACROs which enable and disable attention sources (light pen, alphanumeric keyboard, etc.), request queued attentions, and optionally wait for an attention when nothing is queued.

GAM TELETYPE MODE
The TTY mode is an extension of GAM which allows the graphics station to be used as a control console. GAMTTY allows user reads and writes to the device which all system tasks (PIP, EDI, SYS, etc.) function normally.

GAM TEKTRONIX MODE (GAMTEK)
An optional extension of GAM, GAMTEK allows the GDU to function as a Tektronix (4014) display. The advantage of GAMTEK is that it allows existing software, developed for the 4014, to run on a display supported by GAM.

GRAPHIC SYSTEM USING GAM



P.O. Box 1305/Woodland Hills, California 91365/(213) 346-0549/Twx 910-494-5986

Interactive Graphic Systems Inc.

GDL

GRAPHIC DISPLAY LIBRARY (GDL)
GDL is the application programmer's interface with the display environment. GDL is a library of callable subroutines that provide the graphic programming services to assist the application programmer in creating and manipulating graphic displays. GDL subroutines address one or more Graphic Display Stations (GDSs) attached to any computer supporting GAM. A GDS is defined as a CRT monitor and any combination of interactive devices associated with it.

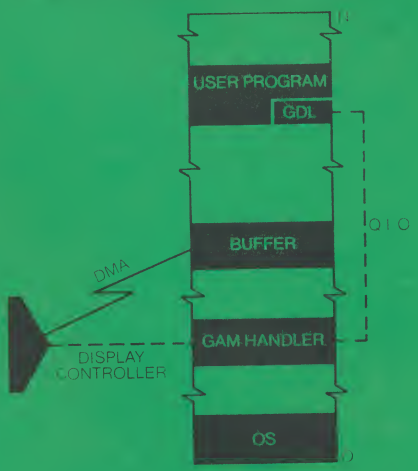
The subroutines in GDL are both modular and open-ended to promote rapid application software development and allow for extending GDL. GDL is designed to relieve the programmer of the responsibilities of handling the graphic station, while allowing him complete access to all of the hardware features.

GDL is a re-entrant resident library and operates on any system that includes IGS's Graphic Access Method (GAM). GDL permits multiple display stations to be controlled from multiple programs. GDL also allows the station to be used as an input device. Inputs can be generated by the alphanumeric keyboard, function keys, light pen, or other input devices. GDL's calling sequences allow subroutines to be called from programs written in FORTRAN, PASCAL, BASIC, CORAL, and Assembly Language.

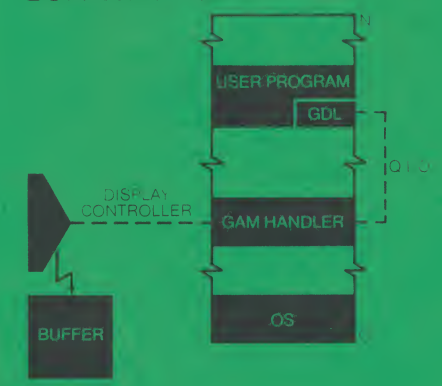
The following GDL features contribute to the display station versatility and ease of use:

- Choice of absolute or relative coordinates which simplifies display manipulations.
- Inhibition of light pen sensing for selected portions of the displayed data: allows use of nonsensed "background" information.

NON-BUFFERED DISPLAY



BUFFERED DISPLAY



- Automatic blinking under program control of any displayed item.
- Multiple levels of display intensification, plus blanking.
- Subroutining capability when information is repeated in different screen positions, which reduces core memory requirements.
- A tracking cross for light pen, data tablet and joystick.

The following programming services are provided by GDL:

- Memory Management
  A dynamic storage allocation routine provides contiguous memory words in various size blocks. Programs request memory when they need it and return it to available space when they no longer need it. Storage allocation, fracturing, and combining are transparent functions to the user.

- Image Structure
  The heart of the GDL system is called a "block". A block is a collection of station orders. The use of blocks allows the programmer to organize display commands into complex structures. A block may be a single contiguous set of memory words obtained from the storage allocator, or a collection of such sets linked together. In either case, the block KEY refers to the entire collection of orders. Compound pictures are formed by connecting blocks to other blocks, yet blocks retain their identity even when they are connected to other blocks.

P.O. Box 1305/Woodland Hills, California 91365/(213) 346-0549/Twx 910-494-5986

Operations are available for creating, combining, and returning blocks to available space through the storage allocator.

Images displayed on the station frequently consist of many parts, some of which must be changed in response to changes in data, requests by the station user, or preplanned dynamic changes. If an image is created by a homogeneous set of station commands, then every change in the image requires reconstruction of the entire command sequence. In multi-user systems, or in cases where changes occur frequently or must be effected rapidly, it is important to avoid unnecessary use of CPU processor time for construction of display images. For this reason, GDL provides two methods for changing parts of pictures:

1. A set of subroutines for naming and changing individual station commands.

2. Subroutines for naming, constructing, and combining parts of pictures in such a way that parts can be altered, replaced, or removed completely from larger parts.

Commands for the station are grouped in blocks. In general, single parts, closely related parts, or parts that change at the same time are grouped into one block. Pictures or compound parts are formed by connecting blocks to other blocks.

- **Modes**
  GDL contains operations for setting modes for light-pen operation, blinking, intensity, and character size.

- **Element Operations**
  GDL contains subroutines for constructing each type of station order and adding it to a specified block. If no space remains in the block when an element subroutine is called, additional storage will be automatically allocated to the block in the quantity specified by the block declaration through the use of memory management.

- **Display Control**
  These commands cause the station to execute orders in a specified block or halt execution.

- **Scaling and Scissoring**
  GDL provides the programmer with a set of routines that allow programming in "user units". The programmer can relate user data to the station screen via the scaling entity. The scaling entity defines the environment that user data is to function through and is related to one or more blocks.

  The scaling entity provides:

  1. Scaling of user data to screen area.
  2. Scissoring at block or screen boundary or no scissoring at all.
  3. Choice of integer or floating point user data.

- **Attention Handling**
  All station interrupts are processed and queued initially by GAM. After GAM has processed an interrupt it is called a graphics attention. GDL subroutines allow programs to request control on specific queued attention-associated data such as that received from function keys or a light pen. The program may optionally specify that GDL is to suspend the program while it waits for an attention.

GFM

GDL FILE MANAGEMENT (GFM)
This optional GDL feature facilitates the translation of image blocks into a (relocatable) form suitable for storage, subsequent retrieval, and retranslation into internal blocks.

P O Box 1305/Woodland Hills, California 91365/(213) 346-0549/Twx 910-494-5986